Referencias

FILLMORE, C. J.: «The Case for Case». En: E. Bach. y R. Harms (Eds.), *Universals in Linguistics Theory*, pp. 1–90. Holt, Rinehart and Winston Publishing Company, 1968.

GÓMEZ, ASUNCIÓN; JURISTO, NATALIA; MONTES, CÉSAR y PAZOS, JUAN.: *Ingeniería del Conocimiento*. Editorial Centro de Estudios Ramón Areces, S.A., 1997.

HENDRIX, G. G.: «Expanding the utility of semantic networks through partitioning». En: *Proceedigns of the IJCAI-75*, pp. 115–121, 1975.

HENDRIX, G. G.: «Encoding Knowledge in Partitioned Networks». En: N. V. Findle (Ed.), *Associative Networks - The Representation and Use of Knowledge in Computers*, pp. 51–92. Academic Press, 1979.

MINSKY, M.: «A Framework for Representing Knowledge». En: R. J. Brachman y H. J. Levesque (Eds.), *Readings in Knowledge Representation*, pp. 245–262. Kaufmann, Los Altos, CA, 1985.

QUILLIAN, M. R.: «Semantic Memory». En: M. Minsky (Ed.), *Semantic Information Processing*, pp. 27–70. The MIT Press, 1968.

TOURETZKY, D.S.: *The Mathematics of Inheritance SystemsThe Mathematics of Inheritance Systems*. Morgan Kaufmann Publishers, Inc, Los Altos (California), EE.UU, 1986.

Capítulo 5

Ontologías

Asunción Gómez Pérez[1], Mariano Fernández López[2] y Óscar Corcho[1]
Universidad Politécnica de Madrid[1] y Universidad San Pablo CEU[2]

5.1 Introducción

Las ontologías empezaron a adquirir relevancia en el área de la informática en 1991, y más concretamente, dentro del *Knowledge Sharing Effort*, campaña emprendida por la agencia norteamericana DARPA (Agencia de Investigación de Proyectos Avanzados de Defensa) [Neches y otros, 1991]. El objetivo de esta campaña era facilitar la construcción de nuevos sistemas, basados en conocimientos, de forma diferente a la que entonces imperaba, para que las bases de conocimientos en las que estos sistemas se fundamentaban no tuvieran que construirse a partir de cero, sino ensamblando los componentes que se podían reutilizar. Esta reutilización sirve tanto para conocimientos estáticos, que se modeliza por medio de ontologías, como para el conocimiento dinámico de resolución de problemas, que se modeliza con PSMs (*Problem Solving Methods*).

Desde 1991 hasta el presente se han producido avances importantes en este área y hoy las ontologías son moneda común que se utiliza en el desarrollo de gran número de aplicaciones en áreas tan diversas como gestión del conocimientos, procesamiento de lenguaje natural, comercio electrónico, integración de información inteligente, recuperación de información, diseño e integración de bases de datos, bioinformática, educación, etc.

La llegada de la Web Semántica [Berners-Lee, 1999] ha hecho ver la necesidad, cada vez mayor, que existe de reutilizar conocimientos a la vez que ha fortalecido su propio potencial, de aquí que las ontologías y los PSMs (a los que, en muchos casos, se les puede considerar precursores de los Servicios Web Semánticos) jueguen un importante papel en este contexto.

Tal y como se describe en el título del artículo, vamos a explicar el *qué* y el *cómo* de las ontologías. En la sección 5.2 definiremos el término "ontología" y explicaremos cuáles son los principales componentes a utilizar cuando se modelizan ontologías. En las siguientes secciones vamos a mostrar cómo se desarrolla una ontología siguiendo METHONTOLOGY y utilizando WebODE (metodología y plataforma propuestas

por el Grupo de Ingeniería Ontológica de la Universidad Politécnica de Madrid). Con esta tecnología nuestro grupo ha construido ontologías para campos tan dispares como el de la química, las ciencias naturales, la gestión del conocimiento, el comercio electrónico, etcétera. El ejemplo que mostramos en este artículo es la adaptación de una taxonomía de clase propuesta por Breuker[1], para construir una ontología de personas jurídicas en el contexto del sistema legal español, para la cual utilizamos METHONTOLOGY (véase la sección 5.3) y WebODE (véase la sección 5.4) como soporte tecnológico. La sección 5.5 describe de forma somera otros métodos y metodologías, así como algunas de las herramientas existentes.

En la sección 5.6 mostraremos brevemente una panorámica de los lenguajes que existen para implementar ontologías, y por último, en la sección 5.7 expondremos las conclusiones. Al final de estas secciones, ofreceremos una lista de enlaces a ontologías y una serie de ejercicios (tanto resueltos como no resueltos)[2].

5.2 Ontologías: definición y componentes

Desde la Antigua Grecia hasta nuestros días se ha estudiado *ontología*, aquella rama del saber que trata sobre la esencia de las cosas que permanece a través de los cambios. Uno de los frutos de la ontología ha sido las ontologías. Para un filósofo, *una ontología* es "*un sistema particular de categorías sistematizando cierta visión del mundo*" [Guarino, 1998]. Un ejemplo de ontología es la de [Lowe, 2006], que distingue entre entes universales (p. ej. ordenador) y particulares (p. ej. este ordenador), y entre sustancias (p. ej. este ordenador) y modos (p. ej. el color de este ordenador).

Los herederos en informática de las ontologías filosóficas son aquellos artefactos compartibles y reutilizables que tienen que ser desarrollados en un lenguaje comprensible para el ordenador [Gruber, 2006; Studer y otros, 1998]. Esta idea está expresada en la definición aportada por [Studer y otros, 1998]: *una ontología es una especificación formal de una conceptualización compartida.* Para nosotros, esta definición es, sin duda, la más completa de las encontradas en la literatura.

Existen diferentes formalismos de representación de conocimientos para formalizar (e implementar) ontologías y cada uno de ellos tiene distintos componentes que pueden ser utilizados en estas tareas de formalización e implementación, sin embargo, dichos formalismos comparten un conjunto mínimo de componentes, a saber:

- **Clases**, que representan conceptos tomados en su sentido más amplio. En el dominio del los viajes, por ejemplo, los conceptos que tenemos son: lugares (ciudades, pueblos, etc.), alojamiento (hoteles, campings, etc.) y medios de transporte (aviones, trenes, coches, transbordadores, motos y barcos). En la ontología, las clases están normalmente organizadas en taxonomías por medio de las cuales se pueden aplicar los mecanismos de herencia. Podemos representar, por ejemplo, una taxonomía de lugares de diversión (teatro, cine, sala de conciertos, etc.)

[1] http://zeus.ics.forth.gr/forth/ics/isl/projects/ontoweb/notes/legal-ontol-ontoweb-sard-2002.ppt

[2] Para profundizar en el campo de la ingeniería ontológica recomendamos el libro Gómez-Pérez y otros [Gómez-Pérez y otros, 2003].

o de viajes organizados (billete clase turista, billete clase preferente, etc.) P otro lado, las metaclases también se pueden definir en el paradigma KR basa en marcos. Metaclases son clases cuyas instancias son también clases y permit hacer una gradación del significado ya que establecen diferentes capas de clas en la ontología en la que están definidas.

- **Relaciones**, que representan un tipo de asociación entre los conceptos del d minio y se definen formalmente como cualquier subconjunto de un producto n conjuntos, es decir: $R \subset C_1 x C_2 x ... x C_n$. Las ontologías normalmente conti nen relaciones binarias, cuyo primer argumento es conocido como dominio la relación, mientras que el segundo es conocido como rango. Por ejemplo, relación binaria `lugar de llegada` tiene el concepto `viaje` como dominio y concepto `lugar` como rango. Las relaciones pueden instanciarse con conocimie tos del dominio, por ejemplo, para expresar que el vuelo AA7462-Feb-08-20 llega a Seattle tenemos que escribir: (`lugarDeLlegada AA7462-Feb-08-20 Seattle`).

  Las relaciones binarias se utilizan algunas veces para expresar atributos de co ceptos, conocidos como ranuras (*slots*) y estos atributos se distinguen de l relaciones porque su rango es un tipo de datos como, por ejemplo, cadena caracteres, número, etc., mientras que el rango de las relaciones es un concept

- **Axiomas**. Según [Gruber, 1992] sirven para modelizar afirmaciones que s siempre ciertas. En ontologías se utilizan generalmente para representar con cimiento que no se puede definir formalmente a través de otros componente además sirven para verificar la consistencia de la ontología misma o la consi tencia de los conocimientos almacenados en una base de conocimientos, por que son muy útiles para inferir conocimientos nuevos. Un axioma en el domin de los viajes sería: *no es posible viajar de América a Europa en tren.*

- **Instancias**, que se utilizan para representar elementos o individuos en u ontología. Un ejemplo de instancia del concepto `AA7462` es el vuelo `AA7462` q llega a Seattle el 8 de febrero del 2006 y que cuesta 300 (dólares americano euros o cualquier otra moneda).

Las ontologías se pueden formalizar, además de con formalismos y lenguajes cre dos específicamente para representar conocimientos a través de enfoques del cam de la ingeniería del software, tales como el *Unified Modeling Language* (UML) [Rum baugh y otros, 1998] o los diagramas entidad relación (ER) [Chen, 1976].

En este contexto, el equipo del *Object Management Group* (OMG) está trabajand en una especificación que pueda definir los metamodelos de algunos de los tipos d diagramas y lenguajes que se utilizan en la representación de ontologías. Esta espec ficación, que es conocida con el nombre *Ontology Description Model* (ODM), utiliz para describir metamodelos una notación formal. Los metamodelos (que ya han sid definidos tanto para UML como para el entidad relación, OWL, RDF(S) etc.) pu den considerarse formalizaciones de ontologías de representación del conocimient

Las correspondencias entre metamodelos se han descrito formalmente en el documento de ODM [OMG, 2005].

El objetivo de dicho documento es hacer que los ingenieros de software puedan modelar ontologías con notaciones que les sean conocidas como, por ejemplo, UML y ER, y transformar sus modelos conceptuales en ontologías formales que se puedan representar en lenguajes de ontologías.

5.3 Una metodología para el desarrollo de ontologías: METHONTOLOGY

METHONTOLOGY [Corcho y otros, 2007; Fernández-López y otros, 1997, 1999] es una metodología diseñada por el Grupo de Ingeniería Ontológica de la Universidad Politécnica de Madrid (UPM) que permite construir ontologías en el nivel de conocimientos, y que se elaboró partiendo de las principales actividades identificadas en el proceso de desarrollo de software de IEEE [IEEE, 1996], y de otras metodologías de ingeniería del conocimiento [Gómez-Pérez y otros, 1997].

Las herramientas ODE y WebODE [Arpírez y otros, 2003] fueron creadas para dar apoyo tecnológico a METHONTOLOGY; sin embargo, existen otras que también se pueden utilizar para construir ontologías siguiendo este método como son: Protégé-2000 [Noy y otros, 2000], OntoEdit [Benton y Kambhampati, 2002], KAON [Maedche y otros, 2003], etcétera. Nuestra metodología, METHONTOLOGY, ha sido recomendada por la *Foundation for Intelligent Physical Agents* (FIPA)[3] para construir ontologías. Esta organización aboga por la interoperabilidad de las aplicaciones basadas en agentes. METHONTOLOGY propone un proceso de desarrollo, un ciclo de vida de la ontología y una especificación para cada una de las actividades que se realizan durante el proceso de desarrollo.

En las siguientes secciones vamos a describir qué entendemos por proceso de desarrollo de ontologías y qué por su ciclo de vida. Después vamos a mostrar el proceso que hemos seguido para conceptualizar una ontología de personas jurídicas (personas jurídicas, organizaciones, etc.) en el dominio del sistema legal español. Como ya hemos comentado previamente, hemos adaptado la taxonomía de clases de personas jurídicas propuesta por Breuker[4] al sistema legal español. Las definiciones que ofrecemos para algunos de los términos legales de nuestra ontología son adaptaciones del lexicón de LEGAMedia[5].

5.3.1 Proceso de desarrollo de ontologías y su ciclo de vida

El proceso de desarrollo de ontologías y su ciclo vida ya fue dado a conocer por [Fernández-López y otros, 1997] en el marco de METHONTOLOGY, y sus propuestas estaban basadas en el estándar que IEEE establece para el desarrollo de software [IEEE, 1996]. Dicho proceso supone llevar a cabo las actividades necesarias para

[3] http://www.fipa.org/specs/fipa00086/

[4] http://zeus.ics.forth.gr/forth/ics/isl/projects/ontoweb/notes/legal-ontol-ontoweb-sard-2002.ppt

[5] http://www.legamedia.net/lx/lx.php

construir ontologías y que pueden clasificarse en tres categorías (véase la Figura 5.1 Actividades de gestión, desarrollo y apoyo.

Las **actividades de gestión** incluyen una *planificación*, así como un *control* una *garantía de calidad*. La actividad del planificación indica las tareas que hay q realizar, y en qué orden, así como el tiempo y los recursos que se necesitan pa completarlas. Esta actividad es esencial para ontologías que utilizan a su vez ontologí almacenadas en bibliotecas de ontologías, o para las que requieren una gran cantid de abstracción y de generalidad. La actividad de *control* garantiza que las tare programadas se terminen de la forma deseada y la actividad de garantía de calid nos asegura que la calidad de todos y cada uno de los productos construidos (ontologí programas y documentación) es satisfactoria.

Las actividades orientadas al desarrollo, tal y como aparecen en la Fig ra 5.1, se pueden clasificar en tres grupos: actividades previas al desarrollo, desarrol propiamente dicho y actividades posteriores. Durante las actividades previas al des rrollo, un *estudio del entorno* identifica el problema a resolver con la ontología, l aplicaciones en donde la ontología tiene que integrarse, etc. y también se lleva a cab un *estudio de viabilidad* que responde a preguntas tales como: ¿es posible constru una ontología?, ¿es conveniente construirla?, etc.

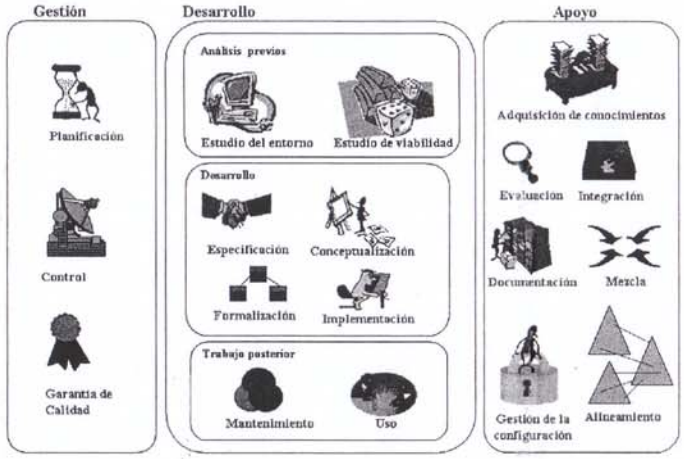

Figura 5.1: Proceso de desarrollo de ontologías (adaptado de [Fernández-López y otro 1997]).

Durante las actividades de desarrollo propiamente dicho, la tarea de *especificació* establece la razón por la cual se construye la ontología, qué usos se quiere que tenga quiénes son los usuarios finales, mientras que la tarea de *conceptualización* estructur

el conocimiento del dominio como modelos con significado bien partiendo de cero, bien reutilizando modelos ya existentes; en este último caso, se realizan actividades tales como recortar ramas de las taxonomías existentes, ampliar la cobertura de las ontologías añadiéndoles nuevos conceptos a los niveles más altos de sus taxonomías, o especializar las ramas que requieren mayor granularidad. Dicha actividad es independiente del lenguaje de implementación. Por otro lado, la tarea de formalización consiste en transformar el modelo conceptual en un modelo formal o semicomputable; por último, la *actividad de implementación* crea modelos computables en un lenguaje de ontologías.

En el periodo posterior al desarrollo, la actividad de *mantenimiento* consiste en actualizar y corregir la ontología si fuera necesario; en este periodo la ontología puede reutilizarse por otras ontologías o aplicaciones. La actividad de *evolución* consiste en controlar los cambios que se producen en la ontología y sus efectos creando y ocupándose de las diferentes variantes de la ontología y teniendo en cuenta que dichas variantes pueden utilizarse en distintas ontologías y aplicaciones. [Noy, 2006].

Por ultimo, **las actividades de apoyo** incluyen una serie de tareas que se pueden realizar durante las actividades dedicadas al desarrollo y sin las cuales la ontología no se podría construir. En estas tareas se incluyen la de *adquisición de conocimientos*, *evaluación*, *integración*, *mezcla*, *alineamiento*, *documentación* y *gestión* o *control de la configuración*. El objetivo de la tarea de *adquisición de conocimientos* es adquirir los conocimientos que los distintos expertos tienen sobre un determinados dominio o adquirir conocimientos a través de algún tipo de proceso semiautomático y que llamamos "aprendizaje de ontologías" [Kietz y otros, 2000]. La actividad de *evaluación* [Gómez-Pérez, 1994] consiste en emitir un juicio técnico sobre las ontologías, sobre sus entornos de software asociados y sobre su documentación. Dicho juicio se hace respecto a un marco de referencia, tanto durante cada fase como entre las fases del ciclo de vida de la ontología. Por otro lado, la actividad de *integración* es necesaria cuando se construye una ontología reutilizando otras ontologías disponibles. Otra actividad de apoyo que hay que mencionar es la de *mezcla* [Gangemi y otros, 1999; Noy y Musen, 2000; Steve y otros, 1998; Stumme y Maedche, 2001], que consiste en extraer una nueva ontología de otras ontologías del mismo dominio. La ontología así obtenida puede unificar conceptos, terminología, definiciones, restricciones, etc, de las ontologías fuente. La mezcla de dos o más ontologías se puede realizar bien durante el tiempo de ejecución, bien durante el diseño. Durante la actividad de *alineamiento* se establecen diferentes tipos de correspondencias entre las ontologías implicadas, de tal forma que las ontologías originales no se fusionan. La actividad de *documentación* detalla de forma clara y exhaustiva cada uno de los estadios completados, así como los productos creados. En cuanto a la actividad de *gestión de la configuración*, diremos que consiste en registrar todas las versiones de la documentación y del código de la ontología con objeto de controlar los cambios. También se puede tener en cuenta una *actividad multilingüe* que consiste en establecer correspondencias entre ontologías y descripciones formales de conocimientos lingüísticos [Declerck y Uszkoreit, 2003]; esta actividad no es siempre considerada una actividad de apoyo, sin embargo, es muy pertinente en el contexto de las ontologías en red que están disponibles en la Web Semántica.

El proceso de desarrollo de ontologías no explica en qué orden hay que realizar l actividades ya que éste es el cometido del **ciclo de vida de la ontología**, el cu se encarga de expresar *cuándo* se tienen que hacer, es decir establece el conjunto c etapas o fases que atraviesa la ontología durante su vida útil, y describe las actividad que hay que realizar en cada fase y la relación entre las fases (relación de precedenci regreso o retorno, etc.)

La primera versión del modelo del proceso del ciclo vida de METHONTOLOG (véase la Figura 5.2) propone comenzar con una programación de las actividades realizar. Y después sigue con la actividad de *especificación*, que consiste en explic por qué hay que construir la ontología, qué posibles usos tendrá, y quiénes serán s usuarios. Cuando la tarea de especificación termina, comienza la de conceptualizació cuyo objetivo es organizar y estructurar los conocimientos adquiridos, utilizando pa ello un conjunto de representaciones que los expertos del dominio pueden manipul fácilmente. Después de que se ha construido el modelo conceptual éste tiene q ser formalizado e implementado (pero si el modelo conceptual es lo suficientemen formal no será necesario pasar por estas dos fases, sino que se puede ir directamen a la implementación). Para ver más detalles, consúltese [Gómez-Pérez y otros, 2003

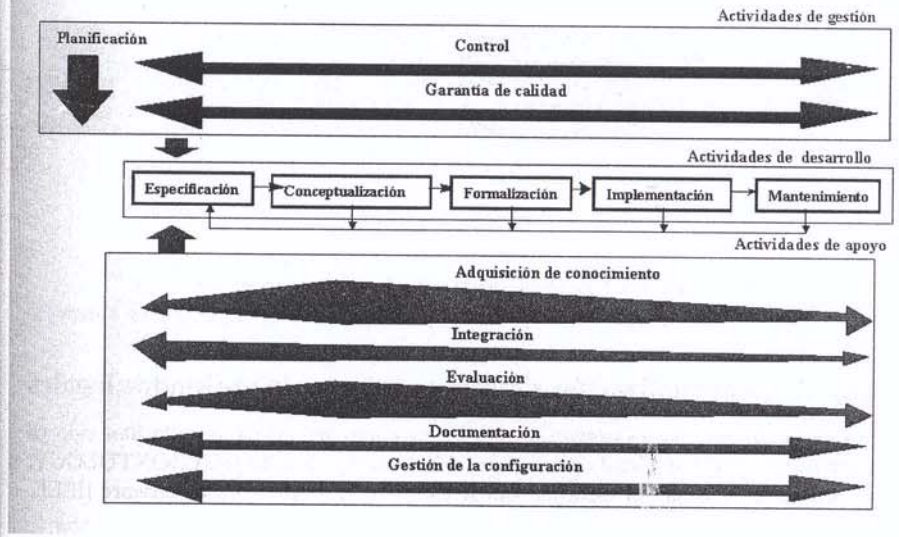

Figura 5.2: Ciclo de vida de la ontología en METHONTOLOGY.

El ciclo de vida de la Figura 5.2 ha sido modificado, ya que existe un gran núme de ontologías disponibles tanto en bibliotecas de ontologías como en Internet, por l que su reutilización por otras ontologías y aplicaciones ha aumentado. Las ontología de dominio pueden ser reutilizadas para construir otras de mayor granularidad cobertura y además se pueden mezclar con otras para crear unas nuevas. Haciend

una analogía con el mapa de un metro subterráneo, podemos observar que existe una línea principal (en el centro de la Figura 5.3) que propone las principales actividades de desarrollo ya señaladas en las anteriores versiones de METHONTOLOGY. Hay otras líneas que nacen de la principal o terminan allí y otras van en paralelo y se bifurcan en un punto. Es por esto que surgen relaciones de interdependencia [Gómez-Pérez y Rojas, 1999] entre el ciclo de vida de varias ontologías, por lo que las acciones de evaluación, recorte y mezcla se pueden realizan en dichas ontologías, es decir, los ciclos de vida de las diferentes ontologías se cruzan, lo que produce diferentes escenarios con diferentes requisitos tecnológicos. [Corcho y otros, 2007] han descrito algunos de los escenarios más comunes que suelen aparecer en este contexto.

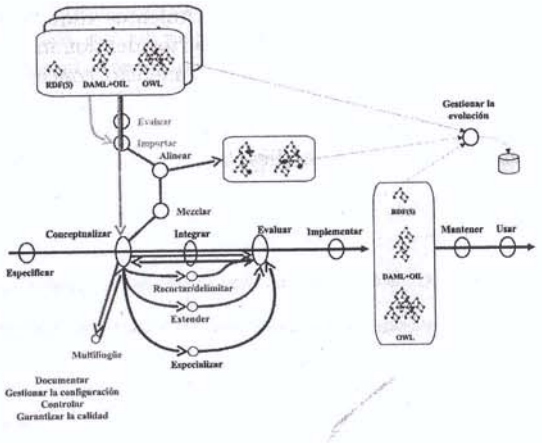

Figura 5.3: Proceso de desarrollo de ontologías en red.

5.3.2 Conceptualización de una ontología de entidades legales

Tanto el proceso para desarrollar ontologías como sus ciclos de vida han sido ya identificados por [Fernández-López y otros, 1997] en el marco de METHONTOLOGY. Sus propuestas se basan el estándar del IEEE para el desarrollo de software [IEEE, 1996].

Al construir ontologías, durante la fase conceptualización el ingeniero no debe ser anárquico en el uso de los componentes de modelización arriba señalados, ni debe definir, por ejemplo, una relación si los conceptos conectados no están definidos en la ontología de forma precisa. METHONTOLOGY incluye en esta fase el conjunto de tareas de conocimiento estructurado que aparece en la Figura 5.4. En la figura se destacan los componentes de la ontología (conceptos, atributos, relaciones, constantes, axiomas formales, reglas e instancias) construidos dentro de cada tarea y muestra el orden que se ha propuesto para crear tales componentes durante la actividad de

conceptualización. Este proceso de modelización no es secuencial aunque haya que seguir un cierto orden para asegurar la consistencia y completitud del conocimiento representado. Si se introduce nuevo vocabulario el ingeniero puede volver a la tarea anterior.

Tarea 1: Construir el glosario de términos. Se debe primeramente construir un glosario de términos que incluya todos los términos relevantes del dominio (conceptos, instancias, atributos, relaciones entre conceptos, etc.), las descripciones de los términos para el lenguaje natural, así como sus antónimos y sinónimos. La Tabla 5.1 muestra una sección del glosario de términos de la ontología de entidades legales. Es importante comentar aquí que en las primeras fases de la conceptualización, el glosario de términos puede contener varios términos que hacen referencia al mismo componente; en ese caso, hay que hacerlos figurar como sinónimos.

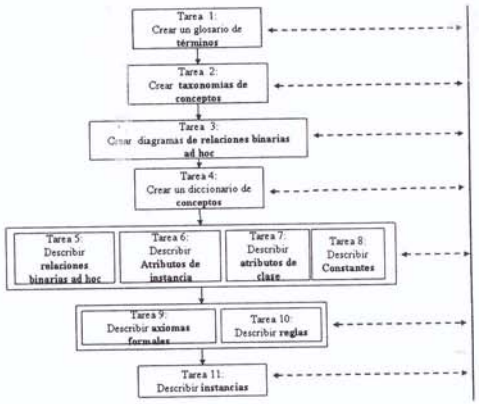

Figura 5.4: Tareas de la actividad de conceptualización según METHONTOLOGY.

Término	*Sinónimos*	*Acrónimos*	*Descripción*	*Tipo*
Mayoría de edad en España	– –	– –	La mayoría de edad en España es de 18 años	Constant
Tribunal	Tribunal de justicia	– – – –	Aunque tribunal puede ser entendido como un lugar físico, asumimos en esta ontología que se trata un conjunto de personas.	Concept
Fecha de nacimiento	– –	– –	El día que nació una persona.	Instance Attribute
Abogado defensor	– –	– –	Quien lleva la defensa en un pleito.	Relation

Tabla 5.1: Fragmento del glosario de términos de la ontología de entidades legales.

Tarea 2: Construir taxonomías de conceptos. Cuando el glosario contiene un número de términos considerable, hay que construir taxonomías de conceptos para

establecer una jerarquía; para ello, se seleccionan del glosario de términos los términos que son conceptos. METHONTOLOGY propone utilizar las cuatro relaciones taxonómicas definidas en la *Frame Ontology* [Farquhar y otros, 1997] y en la Ontología OKBC, a saber: subclase de, descomposición disjunta y descomposición exhaustiva.

Un concepto C_1 es una subclase de otro concepto C_2 si y sólo si cada instancia de C_1 es también una instancia de C_2. En la Figura 5.5 podemos ver cómo `persona física` es una subclase de `persona`, ya que cada persona física es una persona. Un concepto puede ser una subclase de más de un concepto de la taxonomía, por ejemplo, el concepto `empresa de control compartido` es una subclase de los conceptos `empresa privada` y `empresa pública`, puesto que una compañía cuya dirección está compartida puede estar dirigida por entidades tanto públicas como privadas.

Una *descomposición disjunta* de un concepto C es un conjunto de subclases de C que no tienen instancias comunes y que no tienen por qué cubrir C, es decir, puede haber instancias del concepto C que no son instancias de ningún concepto en la descomposición. Por ejemplo (véase la Figura 5.5) los conceptos `ministerio` y `tribunal` forman una descomposición disjunta del concepto `organización` ya que ninguna organización puede ser a la vez un ministerio y un tribunal. Además, puede haber instancias del concepto `organización` que no son instancias de ninguna de las dos clases.

Una *descomposición exhaustiva* de un concepto C es un conjunto de subclases de C que cubren C, y que pueden tener instancias y subclases comunes, es decir, no puede haber instancias del concepto C que no sean instancias de por lo menos uno de los conceptos de la descomposición. Por ejemplo (véase la Figura 5.5) los conceptos `empresa privada` y `empresa pública` forman una descomposición exhaustiva del concepto `empresa` porque no hay compañías que no sean instancias de por lo menos uno de dichos conceptos, aunque pueden tener instancias comunes. Por ejemplo, una `empresa de control compartido` es una `empresa pública` y una `empresa privada`.

Una *partición* de un concepto C es un conjunto de subclases de C que no tienen ninguna instancia en común y que cubre C, es decir, no hay instancias de C que no sean instancias de uno de los conceptos en la partición. Por ejemplo, la Figura 5.5 muestra que los conceptos `menor de edad` y `mayor de edad` forman una partición del concepto `persona física`, porque cada persona física es o menor o mayor de edad. Después de haber estructurado la taxonomía de conceptos y antes de seguir con la especificación de nuevos conocimientos, se deberá comprobar que las taxonomías no contienen errores [Gómez-Pérez, 2006]. Por ejemplo, se deberá comprobar que un elemento no es simultáneamente una instancia de dos clases de una descomposición disjunta, que no hay bucles en la taxonomía de conceptos, que varios términos no tienen el mismo significado, etc.

Tarea 3: Construir diagramas de relaciones binarias *ad hoc*. Una vez que se ha construido y evaluado la taxonomía, hay que construir diagramas de relaciones binarias ***ad hoc*** tal y como propone la actividad de conceptualización. El objetivo de este diagrama es establecer relaciones ***ad hoc*** entre conceptos de la misma (o diferente) taxonomía de conceptos. En la Figura 5.6 podemos ver un fragmento del diagrama de relaciones binarias ***ad hoc*** de nuestra ontología de entidades legales con la relación `es demandante`, `es demandado` y `ve`, así como sus contrarios `tiene`

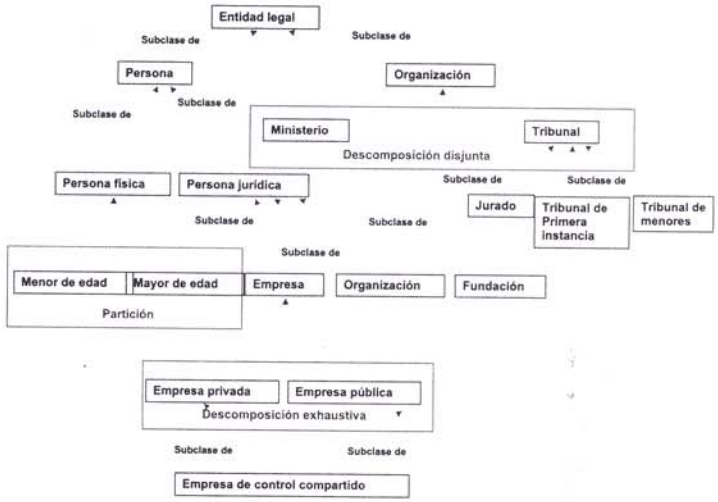

Figura 5.5: Una parte de la taxonomía de conceptos de una ontología de entidades legales.

`demandante`, `tiene demandado` y `es visto`. Tales relaciones conectan los conceptos origen (`persona` y `pleito`; `tribunal` y `pleito`) de las taxonomías de conceptos de entidades jurídicas y pleitos. Desde la perspectiva de la integración de la ontología, dichas relaciones ***ad hoc*** significan que la ontología de entidades legales incluirá la ontología de pleitos y viceversa.

Antes de seguir adelante con la especificación de nuevos conocimientos, se deberá comprobar que el diagrama de relaciones binarias **ad hoc** no contenga ningún error y descubrir si los dominios y rangos de cada argumento de cada relación delimitan de forma precisa y exacta las clases que son apropiadas para la relación. Hay que tener en cuenta que los errores aparecen cuando los dominios y los rangos son imprecisos o excesivamente concretos.

Tarea 4: Construir el diccionario de conceptos. Una vez que se han creado las taxonomías de conceptos y los diagramas de relaciones binarias *ad hoc*, se debe especificar cuáles son las propiedades y las relaciones que describen cada concepto de la taxonomía en un diccionario de conceptos y, opcionalmente, también pueden aparecer sus instancias.

Un diccionario de conceptos contiene todos los conceptos del dominio, sus relaciones, sus instancias, así como sus atributos de clase e instancia. Las relaciones que se especifican para cada concepto son las que tienen por dominio el concepto. Por ejemplo, el concepto `persona` tiene dos relaciones: `es demandante` y `es demandado`. Las relaciones y los atributos de instancia y de clase pertenecen a los conceptos, lo que significa que sus nombres se pueden repetir en diferentes conceptos. La Tabla 5.2

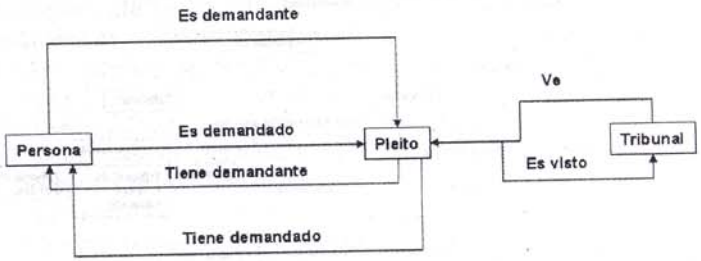

Figura 5.6: Parte del diagrama de relaciones binarias *ad hoc* de la ontología de entidades legales.

muestra una pequeña sección del diccionario de conceptos de la ontología de entidades legales.

Como ya hemos comentado, después de construir el diccionario, se debe describir minuciosamente cada una de las relaciones binarias *ad hoc*, los atributos de clase y los atributos de instancia que aparecen en él y, además, hay que describir de forma pormenorizada cada una de las constantes que aparecen en el glosario de términos. Si bien METHONTOLOGY establece cómo realizar estas tareas, no propone, sin embargo, el orden en el que deben hacerse.

Tarea 5: Definir minuciosamente las relaciones binarias *ad hoc*. El objetivo de esta tarea es describir todas las relaciones binarias *ad hoc* incluidas en el diccionario de conceptos y producir la tabla de relaciones binarias *ad hoc*. Para cada relación binaria *ad hoc*, hay que especificar su nombre, así como los nombres de los conceptos origen y destino, su cardinalidad y su relación inversa. La Tabla 5.3 muestra una sección de la relación binaria *ad hoc* de la ontología de las entidades legales que contiene la definición de las relaciones `es demandado`, `es demandante`, etc.

Tarea 6: Definir detalladamente los atributos de instancia. El objetivo de esta tarea es describir todos los atributos de instancia ya incluidos en el diccionario de conceptos. Cada fila de la tabla de atributos de instancia contiene una descripción pormenorizada de un atributo de instancia. Los atributos de instancia describen las instancias del concepto y su(s) valor(es) pueden ser diferentes para cada instancia del concepto. Para cada atributo de instancia tenemos que especificar los siguientes campos: el nombre, el concepto al que pertenece (los atributos están en o pertenecen a los conceptos), el tipo de valor, el intervalo de valores (en el caso de valores numéricos), la cardinalidad mínima y máxima, los atributos de instancia, los atributos de clase y constantes que se han utilizado para inferir los valores del atributo, los atributos que se pueden deducir con valores de este atributo, las fórmulas o reglas que permitan inferir valores del atributo y las referencias utilizadas para definir el atributo. La Tabla 5.4 muestra un fragmento de los atributos de instancia de la ontología de entidades legales. Algunos de los campos mencionados anteriormente no aparecen por cuestiones

Tabla 5.2: Parte del diccionario de conceptos de la ontología de entidades legales.

Concepto	*Instancias*	*Atributos de Clase*	*Atributos de Instancia*	*Relaciones*
Tribunal	Tribunal de la Audiencia Nacional Tribunal Constitucional Tribunal Supremo Tribunal Provincial de Albacete	–	Número de miembros Sede Jurisdicción territorial	Ve
Empresa	–	Tipo de control	Nombre	–
Abogado defensor	–	–	–	–
Persona	–	–	–	Es demandado Es demandante
Persona física	–	–	Edad Fecha de nacimiento Fecha de muerte Nombre Apellidos Nacionalidad	–

Relación	*Concepto Origen*	*Cardinalidad (Max)*	*Concpeto Destino*	*Relación Inversa*
Es demandado	Person	N	pleito	Tiene demandado
Es demandante	Person	N	pleito	Tiene demandante
Ve	Court	N	pleito	es visto

Tabla 5.3: Parte de la tabla de relaciones binarias *ad hoc* de la ontología de entidades legales.

de espacio. Dicha tabla contiene algunos de los atributos de instancia del concepto `tribunal`: `número de miembros`, `sede` y `jurisdicción territorial`.

El haber utilizado unidades de medida para los atributos numéricos da como resultado la integración de la ontología *Standard Units*. Esto es un ejemplo de cómo METHONTOLOGY propone integrar ontologías durante la actividad de conceptualización, en vez de posponer dicha integración a la actividad de implementación.

Atributo de Instancia	*Concpeto*	*Tipo*	*Dominio Valor*	*Cardinalidad*
Número de miembros	Tribunal	Integer	1 ..	(1, 1)
Sede	Tribunal	String	–	(1, 1)
Jurisdicción territorial	Tribunal	String	–	(1, 1)

Tabla 5.4: Parte de la tabla de atributos de instancia de la ontología de entidades legales.

Tarea 7: Definir minuciosamente los atributos de clase. El objetivo de esta tarea es describir los atributos de clase que ya están incluidos en el diccionario de conceptos usando una tabla de atributos de clase. Cada fila de la tabla de los atributos de clase contiene su descripción detallada. Para cada atributo de clase hay que incluir la siguiente información: el nombre, el nombre del concepto en el que se define el atributo, tipo de valor, valor(es), cardinalidad, los atributos de instancia cuyos valores pueden inferirse con el valor de este atributo de clase, etc. Por ejemplo, el atributo de clase, `tipo de control`, estaría definido por los conceptos de `empresa privada` y `empresa pública`, tal y como aparecen en la Tabla 5.5.

Atributo de Clase	*Concepto*	*Tipo*	*Cardinalidad*	*Valores*
Tipo de control	Empresa privada	[privado,público]	(1,2)	Privado
Tipo de control	Empresa pública	[privado,público]	(1,2)	Público

Tabla 5.5: Parte de la tabla de atributos de clase de la ontología de entidades legales.

Tarea 8: Definir las constantes minuciosamente. El objetivo de esta tarea es describir cada una de las constantes definidas en el glosario de términos. Cada fila de la tabla de constantes contiene una detallada descripción de una constante, y para cada constante hay que especificar el nombre, el tipo de valor (un número, una masa, etc.), el valor, la unidad de medida para las constantes numéricas, así como los atributos que se pueden inferir con la constante. La Tabla 5.6 muestra un fragmento de la tabla de constantes de nuestra ontología de entidades jurídicas, en donde se define

la constante `mayoría de edad en España` y en donde se han omitido los atributos que se pueden inferir con la constante.

Constante	*Tipo*	*Valor*	*Unidad de Medida*
Mayoría de edad en España	Cardinal	18	año

Tabla 5.6: Parte de la tabla de constantes de la ontología de entidades legales.

METHONTOLOGY propone describir axiomas formales y reglas en paralelo después de que se hayan definido los conceptos y sus taxonomías, las relaciones **ad hoc**, los atributos y las constantes.

Tarea 9: Definir axiomas formales. Para realizar esta tarea primero hay que identificar los axiomas formales que la ontología necesita y después describirlos detalladamente. Para cada definición formal de un axioma, METHONTOLOGY propone determinar claramente la siguiente información: el nombre, la descripción en lenguaje natural, la expresión que describe formalmente el axioma con lógica de primer orden, los conceptos, los atributos y las relaciones *ad hoc* a las que el axioma se refiere y las variables que se han utilizado.

Como ya se ha comentado, METHONTOLOGY propone establecer axiomas formales en lógica de primer orden. La Tabla 5.7 muestra un axioma formal de nuestra ontología de entidades legales que estipula que una persona no puede ser demandante y demandado en el mismo pleito. Las columnas que corresponden a los conceptos ya referidos y a las relaciones contienen los conceptos y las relaciones que se utilizan en el axioma formal. Las variables utilizadas son `?X` para `persona` e `?Y` para `pleito`.

Hemos de comentar que la definición de la expresión lógica puede resultar difícil para un experto que no tenga experiencia en lógica de primer orden.

Tarea 10: Definir las reglas. Al igual que en la tarea anterior, hay que identificar primero qué reglas se necesitan en la ontología, y luego describirlas en la tabla de reglas. Para definir cada regla, METHONTOLOGY propone incluir la siguiente información: nombre, descripción en lenguaje natural, la expresión que describe formalmente la regla, los conceptos, atributos y relaciones a los que la regla se refiere, y las variables que se utilizan en la expresión.

Además, METHONTOLOGY propone especificar las expresiones de reglas por medio de la plantilla *if* <*condiciones*>*then* <*consecuente*>. El lado izquierdo de la regla está formado por conjunciones de átomos mientras que el lado derecho está formado por un solo átomo.

La Tabla 5.8 muestra una regla que estipula que los juicios en los que los demandados tienen 14 años o menos, deben celebrarse en un tribunal de menores. Esta regla nos permite inferir el tipo de tribunal. Como aparece en la tabla, la regla hace referencia a los conceptos `menor de edad`, `pleito` y `tribunal`, al atributo `edad`, y a las relaciones `es demandado` y `ve`, y las variables empleadas son ?X para `menor de edad`, ?Y para `integer`, `pleito` para `?Z` y ?Z para `tribunal`.

Al igual que en el caso de los axiomas formales, definir la expresión de la regla puede resultar difícil para los expertos con poca experiencia en lógica de primer orden.

Regla	*Descripción*	*Expresión*	*Conceptos*	*Atributos Afectados*	*Relaciones Afectadas*	*Variables*
Tribunal de menores para menores	Los juicios en los que los demandados tienen 14 años o menos, deben celebrarse en un tribunal de menores	If [menor de edad](?X) and pleito(?Z) and tribunal(?W) and edad(?X, ?Y) and ?Y >14 and [es demandado](?X, ?Z) and ve(?W, ?Z) then [tribunal de menores](?W)]	Menor de edad Pleito Tribunal	Edad	Es demandado Ve	?X ?Z ?W

Tabla 5.8: Parte de la tabla de reglas de la ontología de entidades legales.

Axioma	*Descripción*	*Expresión*	*Conceptos Asociados*	*Relaciones Asociadas*	*Variables*
Incompatibilidad demandante-demandado	Una persona no puede ser demandante y demandado en el mismo pleito	not (exists (?X,?Y) (persona (?X) and pleito (?Y) and [es demandante] (?X,?Y) and [es demandado] (?X,?Y)))	persona pleito	Es demandante, Es demandado	?X ?Y

Tabla 5.7: Parte de la tabla de axiomas de la ontología de entidades legales.

Tarea 11: Definir instancias. Después de crear el modelo conceptual de la ontología, hay que definir instancias relevantes que aparecerán en el diccionario de conceptos y dentro de una tabla de instancias. Para cada instancia hay que definir lo siguiente: su nombre, el nombre del concepto al que pertenece, y sus valores en los atributos, si se conocen. La Tabla 5.9 muestra algunas instancias de la tabla de instancias de nuestra ontología de entidades legales: `Audiencia Nacional`, `Tribunal Supremo` y `Tribunal Constitucional`), que son todas ellas instancias del concepto `tribunal`, tal y como aparece definido en el diccionario de conceptos, y tienen algunos valores de atributo y relación especificadas para : `sede`, `jurisdicción territorial`, y `número de miembros`. Dichas instancias pueden tener más de un valor para los atributos cuya cardinalidad máxima es superior a uno.

Instancia	*Concepto*	*Attributo*	*Valores*
Tribunal de la Audiencia Nacional	Tribunal	Sede	Madrid
		Jurisdicción territorial	España
Tribunal Supremo	Tribunal	Jurisdicción territorial	España
Tribunal Constitucional	Tribunal	Número de miembros	12
		Jurisdicción territorial	España

Tabla 5.9: Parte de la tabla de instancias de la ontología de entidades legales.

Diferentes grupos han utilizado METHONTOLOGY para construir ontologías de química, ciencias naturales, gestión de conocimientos, comercio electrónico, etc. Para una descripción más detallada de esta metodología para construir ontologías ver [Gómez-Pérez y otros, 2003].

5.4 Cómo construir una ontología legal con WebODE

WebODE[6] [Arpírez y otros, 2003] es una plataforma de ingeniería ontológica desarrollada por el Grupo de Ingeniería Ontológica de la Universidad Politécnica de Madrid (UPM) cuya versión actual es la 2.0. WebODE es sucesor directo de ODE (entorno para diseñar ontologías), una herramienta para crear ontologías que se basa en tablas y grafos, y permite a los usuarios personalizar el modelo de conocimientos utilizado para conceptualizar las ontologías según las necesidades que tengan de representación de conocimientos. Tanto ODE como WebODE sirven de apoyo a METHONTOLOGY, la metodología para construir ontologías, que ya hemos descrito en la sección anterior.

La versión actual de WebODE contiene un editor de ontologías, que incluye casi todos los servicios que la plataforma ofrece, además de un sistema de gestión de conocimientos basado en ontologías (ODEKM), un generador automático de portales de la Web Semántica (ODESeW), y una herramienta de edición de servicios, también de la Web Semántica (ODESWS). Para obtener una descripción detallada de cada una de ellas, véase [Gómez-Pérez y otros, 2003].

A continuación, vamos a describir el editor de ontologías de WebODE. Este editor es una aplicación Web que se ha construido sobre el servicio de acceso de la ontología

[6] http://webode.dia.fi.upm.es/

(ODE API), y que para construir ontologías en la plataforma integra varios servicios: edición de la ontología, navegación, documentación, mezcla, razonamiento, etc. En este editor se combinan tres interfaces de usuario, a saber: un editor HTML basado en formularios para editar todos los términos ontológicos menos los axiomas y las reglas, una interfaz gráfica de usuario, llamada ODEDesigner, para editar gráficamente taxonomías de conceptos y relaciones, y WAB (*WebODE Axiom Builder*) para editar axiomas y reglas formales. En esta sección vamos a describir dichas interfaces destacando sus rasgos más importantes.

La Figura 5.7 muestra una captura de pantalla de la interfaz HTML para editar atributos de instancias del concepto `persona física` de nuestra ontología. Las principales áreas de este interfaz son:

- El área de exploración. Para navegar por toda la ontología, crear nuevos elementos y modificar o borrar los que ya existen.
- El portapapeles. Para copiar y pegar fácilmente información entre formularios, y así crear de una forma fácil componentes similares de la ontología.
- El área de edición. Para introducir, borrar y actualizar términos de la ontología (conceptos, atributos, relaciones, etc.) con formularios HTML y tablas con conocimiento sobre los términos existentes.

La Figura 5.7 muestra los atributos definidos para el concepto persona física, que son: `edad`, `fecha de nacimiento`, `fecha de muerte`, etc.

ODEDesigner facilita la construcción de taxonomías de conceptos y de relaciones binarias *ad hoc* entre conceptos, a la vez que permite definir vistas para destacar o personalizar la visualización de fragmentos de la ontología a distintos usuarios.

Las taxonomías de conceptos se crean con las siguientes relaciones conjuntos de relaciones predefinidas: *subclase de*, *descomposición disjunta*, *descomposición exhaustiva*, *partición* y *parte de*. La Figura 5.5 y la Figura 5.6 muestran distintas vistas de nuestra ontología de entidades legales, hechas en ODEDesigner.

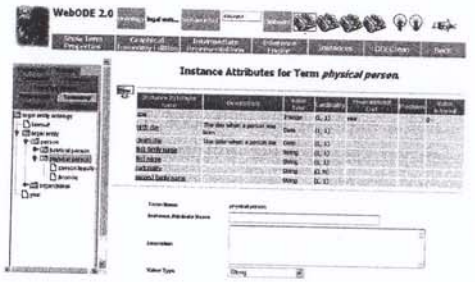

Figura 5.7: Edición de un atributo de instancia con el editor de ontologías WebODE.

El *WebODE Axiom Builder* (WAB) es un editor gráfico para construir axiomas formales y reglas como las que aparecen en la Tabla 5.7 y la Tabla 5.8. Dicho editor

tiene como objetivo facilitar a los expertos en el dominio, y que no tengan mucha experiencia en modelar en lógica de primer orden, la creación de los componentes antes comentados.

A continuación vamos a describir otros servicios para construir ontologías que se encuentran integrados en el editor de la ontología, a saber: el servicio de documentación, ODEMerge, y el servicio de evaluación. Hay muchos otros servicios que no vamos a describir aquí porque creemos que no tienen ninguna utilidad especial para los lectores a los que este libro va dirigido, algunos de estos son: el motor de inferencias de Prolog que está basado en OKBC, ODEClean, los servicios de traducción etc.

El servicio de documentación de ontologías de WebODE genera documentación de ontologías WebODE en diferentes formatos, como son las tablas en HTLM que muestran las representaciones intermedias de METHONTOLOGY, descritas en la sección 5.3, y las taxonomías de conceptos en HTML.

El servicio de mezcla de WebODE (ODEMerge) realiza y supervisa la fusión de conceptos, atributos y relaciones binarias *ad hoc* pertenecientes a dos ontologías construidas para el mismo dominio; este servicio utiliza fuentes de lenguaje natural para encontrar correspondencias entre los componentes de las dos ontologías y así generar una nueva ontología. Finalmente, comentaremos que la plataforma WebODE, ofrece una serie de funciones para evaluar ontologías tales como la que analiza la consistencia de las ontologías o los servicios de evaluación de RDF(S), DAML+OIL, y de OWL.

El servicio que asegura la consistencia de las ontologías ofrece la posibilidad de controlar las restricciones de las ontologías de WebODE y lo utiliza el editor durante el proceso de construcción. Dicho servicio controla las restricciones de tipo, de los valores numéricos y de cardinalidad y verifica las taxonomías de conceptos (por ejemplo, instancias externas de descomposición exhaustiva, bucles, etc.)

Los servicios de evaluación de RDF(S), DAML+OIL y OWL evalúan ontologías siguiendo los criterios de evaluación señalados por [Gómez-Pérez, 2006], y detectan los errores de las ontologías implementadas en los susodichos lenguajes a la vez que sugieren unos criterios mejores para el diseño de las ontologías.

5.5 Otros métodos y herramientas para desarrollar ontologías

En esta sección vamos a explicar los métodos, la metodología y las herramientas que no hemos comentado en los capítulos anteriores.

5.5.1 Metodologías y métodos

Existen en la literatura y desde hace tiempo, una serie de métodos y metodologías para desarrollar ontologías. Ya en 1990, [Lenat y Guha, 1990] publicaron una serie de guías generales e hicieron algunos comentarios interesantes sobre el desarrollo de Cyc; años después, en 1995, [Grüninger y Fox, 1995], propusieron las primeras directrices basándose en la experiencia que habían adquirido al desarrollar la *Enterprise Ontology* [Uschold y Grüninger, 1996] y el proyecto de ontologías TOVE (*TOronto Virtual*

Enterprise) ambas ontologías pertenecientes al dominio de modelado de la empresa [Grüninger y Fox, 1995], que fueron más tarde mejoradas en [Uschold y Grüninger 1996].

En la XII Conferencia Europea de Inteligencia Artificial (ECAI'96) [Bernaras otros, 1996] presentaron, como parte del proyecto Esprit KACTUS, el método utiliza do para construir una ontología en el dominio de las redes eléctricas; la metodologí METHONTOLOGY apareció simultáneamente a aquella y ha seguido ampliándos en artículos sucesivos [Corcho y otros, 2007; Fernández-López y otros, 1997, 1999]. E 1997, se propuso un nuevo método para construir ontologías, basado en la ontologí SENSUS, [Benton y Kambhampati, 1997] y posteriormente, apareció la metodolo gía On-To-Knowledge , desarrollada dentro del proyecto del mismo nombre [Staab otros, 2001].

Dependiendo de los objetivos marcados, podemos utilizar una metodología u otra METHONTOLOGY es la metodología que describe de forma más detallada los pro cesos a realizar. Por otro lado, On-To-Knowledge es la que cubre mayor número d actividades aunque con muy pocas descripciones de los procesos llevados a cabo; e cuanto a la metodología de Grüninger y Fox debemos comentar que es la más for mal de todas y, aunque todas las metodologías cuentan con reutilizar las ontología existentes durante el proceso de desarrollo, sólo METHONTOLOGY ha adaptad recientemente, su propuesta de un ciclo de vida al entorno de las ontologías en re De todas formas, hemos de comentar que las actividades de desarrollo, y sobre tod las de conceptualización e implementación son las que aparecen de forma más deta llada en todas las metodologías. Sin embargo, todavía se echan en falta propuesta para actividades de gestión (planificación, control y garantía de calidad), así como d actividades previas al desarrollo (estudio del entorno) y posteriores al desarrollo (pc ejemplo, la [re]utilización).

Para ampliar el estudio sobre las metodologías para el desarrollo de ontologías s puede consultar [Fernández-López y Gómez-Pérez, 2002].

5.5.2 Herramientas

En cuanto a la tecnología de las herramientas, podemos comentar que ésta h mejorado enormemente desde la creación de los primeros entornos. Si tenemos e cuenta su evolución desde su aparición a mediados de los noventa, podemos distingu dos grupos distintos:

- Herramientas cuyo modelo de conocimiento se corresponde directamente con u lenguaje de implementación de ontologías. Estas herramientas se desarrollaro como editores de ontologías para un lenguaje específico y dentro de este grup podemos incluir a Ontolingua Server [Farquhar y otros, 1997], que respald la construcción de ontologías con Ontolingua y con KIF; a Ontosaurus Serv [Benton y Kambhampati, 1997] con Loom, y a OilEd [Bechhofer y otros, 2001 primero con OIL, luego con DAML+OIL y ahora con OWL.

- Juegos de herramientas integradas cuya principal característica es que tiene una arquitectura extensible y su modelo de conocimientos es generalmente i

dependiente de un lenguaje de ontologías. Además, ofrecen un conjunto básico de servicios relacionados con las ontologías y pueden ampliarse fácilmente con otros módulos para ofrecer más funciones. A este grupo pertenecen Protégé-2000 [Noy y otros, 2000]), WebODE [Arpírez y otros, 2003], OntoEdit [Benton y Kambhampati, 2002] y KAON [Maedche y otros, 2003].

Protégé [Noy y otros, 2000]. Esta herramienta ha sido desarrollada en el *Stanford Medical Informatics* (SMI) de la Universidad de Stanford. Es una aplicación autónoma, de código abierto y con arquitectura extensible. El núcleo de este entorno es el editor y la herramienta tiene una biblioteca de extensiones que le da más funcionalidad al entorno. Normalmente, las extensiones o plug-ins están disponibles para importar/exportar lenguajes de implementación de ontologías (FLogic, Jess, XML, Prolog), diseño de lenguajes, [Knublauch y otros, 2004], acceso a OKBC, creación y ejecución de restricciones (PAL), fusión de ontologías [Noy y Musen, 2000], etc.

OntoEdit [Benton y Kambhampati, 2002]. Es una herramienta desarrollada en la Universidad de Karlsruhe y comercializada por Ontoprise. Es similar a las anteriores, es decir, es un entorno extensible y flexible, basado en una arquitectura de extensiones que ofrece cierta funcionalidad para explorar y editar ontologías, además, incorpora extensiones que se encargan de hacer inferencias utilizando Ontobroker y de exportar e importar ontologías en diferentes formatos (FLogic, XML, RDF(S) y OWL), etc. Hay dos versiones de OntoEdit disponibles: la gratuita, Ontoedit Free, y la profesional, Ontoedit Professional.

El juego de herramientas de **KAON1** [Maedche y otros, 2003] es un entorno de ingeniería ontológica, de código abierto y extensible que define el modelo de conocimientos que subyace en una extensión de RDF(S). OI-modeler es el editor de estas herramientas y el encargado de hacer evolucionar las ontologías, encontrar correspondencias entre ellas y generarlas a partir de las bases de datos, etc.

El IBM Ontology Management System, también conocido por **SNOBASE**, carga ontologías desde ficheros a través de Internet para crear, modificar, consultar y almacenar ontologías localmente. Permite manipular ontologías escritas en RDF Schema y OWL. En la actualidad, SNOBASE respalda una variante de OWL Query Language (OWL-QL) [Fikes y otros, 2003].

Swoop [Kalyanpur y otros, 2006] es una herramienta basada en una arquitectura de extensiones, que respalda la anotación en colaboración a través de Annotea [Koivunen, 2006], y detecta las inconsistencias. Actualmente se trabaja para que Swoop dé soporte a la evolución de ontologías.

Un aspecto interesante de las herramientas es que solamente OntoEdit y WebODE **respaldan metodologías para construir ontologías** (On-To-Knowledge y METHONTOLOGY respectivamente), aunque esto no les impida ser utilizadas con otras metodologías o con ninguna.

Desde la perspectiva del paradigma de representación de conocimientos, las herramientas KAON1 se basan en redes semánticas y en marcos. Por otro lado, WebODE, Protégé, OntoEdit y KAON1 pueden representar conocimientos de acuerdo con una propuesta híbrida que se basa en marcos y en lógica de primer orden. En cuanto a SNOBASE y Swoop, éstas utilizan lógicas descriptivas. Un hecho a tener en cuenta es

la **expresividad del modelo de conocimientos subyacente en la herramienta**. Con todas las herramientas se pueden representar clases, relaciones, atributos e instancias pero sólo KAONI y Protégé cuentan con componentes flexibles como las metaclases. También es importante conocer, antes de escoger una herramienta para desarrollar ontologías, los servicios de inferencias adjuntos a la herramienta como: mecanismos para examinar las restricciones y la consistencia, el tipo de herencia (sencilla, múltiple, monotónica, no-monotónica) clasificaciones automáticas, manejo de las excepciones y ejecución de los procedimientos. KAONI no tiene motor de inferencias, OntoEdit utiliza FLogic [Kifer y otros, 1995] como motor de inferencias; WenODE utiliza Ciao Prolog [Hermenegildo y otros, 2000], Protégé utiliza un motor interno PAL, con SNOBASE se pueden hacer consultas a través de OWL-Ql, y Swoop utiliza un motor de inferencias basado razonadores de lógica de primer orden (por defecto, Pellet). Además Protégé y WebODE ofrecen servicios para evaluar las ontologías y ambos llevan un módulo que realiza las evaluaciones según el método OntoClean [Guarino y Welty, 2002; Welty y Guarino, 2001]. Por último, Protégé (con las extensiones de OWL) realiza clasificaciones automáticas al conectarse a un razonador de lógicas descriptivas.

Otros aspectos importantes de las herramientas a tener en cuenta son **la arquitectura de software y la evaluación de herramientas** (autónoma, cliente/servidor, aplicación de n-hileras/filas) extensibilidad, almacenamiento de las ontologías (bases de datos, ASCII, ficheros, etc.), tolerancia a fallos, gestión de seguridad/copias/respaldo, estabilidad y política sobre las versiones de las herramientas. Desde esta perspectiva, todas estas herramientas basadas en plataformas Java pueden servir para almacenar ontologías en bases de datos. La función para la gestión de seguridad la ofrece WebODE, mientras que KAON, OntoEdit, Protégé, WebODE, SNOBASE y Swoop ofrecen servicios de extensibilidad.

La **interoperabilidad** con otras herramientas, con sistemas de información y con bases de datos, así como la traducción a y desde algunos lenguajes de ontologías son otros rasgos importantes a tener en cuenta al integrar ontologías en aplicaciones. Casi todas las herramientas importan y exportan XML y otros lenguajes de demarcación, y aunque todavía no existe un estudio que compare la calidad de todos estos traductores, tampoco existen resultados empíricos sobre la posibilidad de intercambiar ontologías entre diferentes herramientas ni sobre la cantidad de conocimientos que se pierden en los procesos de traducción. En el taller de EON2003 se dieron a conocer algunos de los trabajos que se están realizando sobre el tema.

En cuanto a **cooperar y colaborar en la construcción de ontologías**, Protégé incorpora algunas funcionalidades de sincronización, Por lo general, las herramientas existentes necesitan tener casi todas las características antes aludidas para que la colaboración en la construcción de ontologías tenga éxito. XML.com tiene un inventario de herramientas para el desarrollo de ontologías[7].

[7] http://www.xml.com/pub/a/2004/07/14/onto.html

5.6 Lenguajes de implementación de ontologías

Los lenguajes de implementación de ontologías se comenzaron a elaborar a principios de los años noventa, y fueron el resultado de la evolución normal de los lenguajes de representación de conocimiento (KR). Básicamente, los paradigmas de representación de conocimientos subyacentes a los lenguajes de ontologías estaban basados en lógica de primer orden (p.ej., KIF [Genesereth y Fikes, 1992], en marcos combinados con dicha lógica, (p.ej., Ontolingua [Farquhar y otros, 1997; Gruber, 1992], OCML [Motta, 1999] y FLogic [Kifer y otros, 1995]), y en lógicas descriptivas (p.ej., Loom [MacGregor, 1991]). En 1997, se creó OKBC [Chaudhri y otros, 1998] para que actuara como un protocolo unificador basado en marcos y acceder a las ontologías implementadas en diferentes lenguajes (Ontolingua, Loom y CycL, entre otras), si bien sólo se utilizó en un reducido número de aplicaciones.

El gran auge de Internet hizo que se crearan lenguajes de implementación de ontologías para poder explotar las características de la Web; a estos lenguajes se les conoce normalmente como lenguajes de ontologías basados en la Web o lenguajes de marcación de las ontologías y su sintaxis se fundamenta en la sintaxis de los lenguajes de marcación existentes como HTML [Raggett y otros, 1999] y XML [Bray y otros, 2000], cuyo objetivo no es el desarrollo de ontologías sino la presentación y el intercambio de datos, respectivamente. Los ejemplos más sobresalientes de dichos lenguajes de demarcación son: SHOE [Luke y Heflin, 2000], XOL [Karp y otros, 1999], RDF [Lassila y Swick, 2000], RDF Schema [Brickley y Guha, 2004], OIL [Horrocks y otros, 2000], DAML+OIL [van Harmelen y otros, 2001] y OWL [Dean y Schreiber, 2004], y de todos ellos sólo RDF, RDF Schema y OWL están recibiendo respaldo de forma activa. Por último, debemos añadir que se está desarrollando una ontología, llamada WSML en el contexto del trabajo que se realiza en el campo de la Web Semántica y más concretamente en el marco del WSMO. A continuación, vamos a comentar los lenguajes de marcado más importantes ya que son muy útiles dentro del contexto de los Servicios Web Semánticos.

RDF [Lassila y Swick, 2000] fue desarrollado por el W3C como un lenguaje basado en la red semántica para describir recursos Web. También RDF Schema [Brickley y Guha, 2004] fue elaborado por el W3C como una extensión de RDF con primitivas basadas en marcos. La combinación de ambos lenguajes, RDF y RDF Schema, RDF(S), y este lenguaje, para el que se han creado motores de inferencia y lenguajes de consulta, sólo sirve para representar conceptos, taxonomías de conceptos y relaciones binarias.

OWL (Ontology Web Language) fue recomendado por el W3C en febrero del 2004. Este lenguaje esta construido encima de RDF(S) y amplía su expresividad con más primitivas, lo que le permite representar expresiones complejas y describir conceptos y relaciones. OWL está dividido en tres capas o estratos (Lite, DL y Full), cada una de las cuales ofrece distintos niveles de expresividad según las necesidades de representación e inferencia de la ontología. Este lenguaje se basa en SHOIN(D+), un lenguaje de lógica descriptiva que cuenta con varios motores de inferencia que se pueden utilizar para examinar las restricciones de conceptos, propiedades e instancias y para clasificar, automáticamente, los conceptos de forma jerárquica.

Por ejemplo, con OWL podemos describir un vuelo como una clase de viaje en el que el medio de transporte utilizado es un avión. Si determinamos esta condición como necesaria y suficiente y luego definimos un viaje en el que se utiliza un avión ligero como medio de transporte (y asumimos que *avión ligero* es una especialización de *avión*), entonces el razonador concluirá que este *viaje* es una especialización de *vuelo*. Este mismo principio se puede utilizar igualmente para examinar la consistencia de la definición que la ontología ofrece.

Y por último, el lenguaje **WSML** (Web Service Modelling Language) [Bruijn, 2006] que se está desarrollando en el marco del WSMO. El objetivo de este lenguaje es que sea utilizado no únicamente en la representación de ontologías sino también en la de los Servicios Web Semánticos, para lo cual cuenta con muchas características adicionales que los lenguajes antes mencionados no tienen. Al igual que OWL, está dividido en capas, cada una de las cuales se basa en distintos formalismos KR, a saber: lógica descriptiva, programación lógica y lógica de primer orden.

De todos estos lenguajes aquí comentados, sólo algunos como Ontolingua, OCML, Flogic, RDF, RDF Schema y OWL (OIL y DAML+OIL también respaldan esta noción pero ya no están activos), están bien equipados con primitivas para explotar el concepto de ontologías en red. Basándonos en nuestra experiencia y en los estudios de los casos disponibles en la literatura, hemos podido encontrar algunas asociaciones entre los lenguajes de ontología y las diferentes clases de aplicaciones basadas en ontologías en las que se aplican.

En las aplicaciones de comercio electrónico, las ontologías se utilizan generalmente para representar productos que se ofrecen en plataformas electrónicas y se muestran a los usuarios a través de catálogos que pueden examinar [Léger y otros, 2000]. Las exigencias de representación de estas ontologías no son extremas, básicamente se necesitan conceptos y atributos y relaciones n-arias entre conceptos; sin embargo, las exigencias de razonamiento son mayores: si el número de productos o servicios que se ofrecen en la plataforma es alto, entonces las clasificaciones automáticas son muy útiles para organizar los productos o servicios (por lo que los lenguajes basados en lógica de descripción son muy, muy útiles) e igual de útil es una eficiente contestación a las consultas (esta rasgo lo tienen casi todos los lenguajes estudiados).

Cuando se utilizan conjuntamente PSMs y ontologías de dominio, hay dos lenguajes que se recomiendan encarecidamente: OCML y FLogic [Fensel, 1995; Motta, 1999], porque respaldan explícitamente la integración y la reutilización de bibliotecas.

Dentro del contexto de la Web Semántica y para poder intercambiar ontologías entre aplicaciones, hay una serie de lenguajes basados en XML que son fáciles de leer y gestionar porque las bibliotecas estándares para el tratamiento de XML están disponibles gratuitamente. Además, no es difícil adaptar los lenguajes tradicionales a la sintaxis de XML y así podrían utilizar el mismo tipo de bibliotecas. El gran mérito de RDF(S) y OWL es el amplio apoyo que reciben de comunidades ajenas a la comunidad ontológica, lo que implica que haya más herramientas disponibles para editar, gestionar y documentar ontologías.

Crear ontologías de nivel superior exige una gran expresividad pero no requiere gran respaldo en el razonamiento. Estas ontologías están normalmente especificadas en lenguajes de lógicas descriptivas como LOOM y CLASSIC.

Por lo general, los lenguajes que se basan en lógicas descriptivas han sido muy utilizados en aplicaciones que requieren integración inteligente o fuentes de información heterogéneas y recogida de información [Barish y otros, 2000; Guarino y otros, 2006; Mena y otros, 1996; Stuckenschmidt, 2000]. Y la razón de que sean tan utilizadas se debe a su soporte de inferencia.

5.7 Resumen

A comienzos de los años noventa, el desarrollo de las ontologías era un arte: los desarrolladores no tenían unas directrices claras para construirlas, sólo contaban con algunos criterios de diseño a seguir. El trabajo que se ha ido haciendo sobre principios, métodos y metodologías además de la tecnología existente ha convertido el desarrollo de ontologías en una ingeniería. Este proceso migratorio se debe principalmente a la definición del proceso de desarrollo de ontologías y de su ciclo de vida, que describe los pasos a seguir para construir ontologías y las interdependencias entre los pasos.

En este capítulo hemos estudiado los componentes de las ontologías, los métodos, metodologías, herramientas y lenguajes que existen en la actualidad y además hemos mostrado con un ejemplo cómo se puede desarrollar una ontología utilizando una metodología y un software en particular: METHONTOLOGY y WebODE. En concreto, hemos mostrado cómo se desarrolla una ontología de entidades legales en el contexto español al adaptar una taxonomía, también de entidades legales, creada por Breuker. Una conclusión evidente que podemos extraer de este ejemplo es que no es necesario tener una gran experiencia en la representación del conocimiento para hacer una ontología. Con METHONTOLOGY podemos modelar ontologías a través de representaciones intermedias tanto gráficas como tabulares que los expertos de cualquier dominio comprenden sin estar necesariamente involucrados en el mundo de las ontologías. Además, WebODE es una plataforma de software que da apoyo a METHONTOLOGY, aunque no fuerza a seguir dicha metodología.

En este capítulo también hemos presentado otros métodos y herramientas para que los lectores puedan trabajar con otras propuestas.

En cuanto a las herramientas, hay que decir que sólo hemos descrito brevemente las que consideramos más relevantes en la actualidad. Curiosamente, sólo WebODE y OntoEdit están pensadas para poder dar soporte a las metodologías METHONTOLOGY y OntoKnowledge, aunque no sólo a ellas. Y añadir que hoy por hoy, el que una herramienta deba ser el reflejo de una metodología no es una idea muy generalizada.

Los lenguajes de ontologías han estado siempre en continua evolución desde que los primeros lenguajes estaban disponibles para implementar ontologías, la mayoría de los cuales estaban fundamentados en los lenguajes que existían de representación de conocimientos.

En los recientes avances habidos en el desarrollo de lenguajes, creados en el contexto de la Web Semántica (RDF, RDF Schema and OWL) y de los Servicios Web Semánticos (WSML), se han tenido muy en cuenta las ontologías en red discrepantes, además, se han añadido espacios de nombres que designan componentes de ontologías definidos en cualquier otro sitio, y se utilizan las primitivas de importación con objeto

de incluir ontologías ya existentes. A continuación, mostramos algunos enlaces a l ontologías más relevantes:

- **ONTOLOGÍAS DE ALTO NIVEL**:
  - DOLCE (http://www.loa-cnr.it/DOLCE.html)
  - SUO (http://suo.ieee.org/)
- **ONTOLOGÍAS LINGÜÍSTICAS**
  - WordNet (wordnet.princeton.edu/)
  - EuroWordnet (http://www.hum.uva.nl/ ewn/)
  - SENSUS (http://www.hum.uva.nl/ ewn/)
- **ONTOLOGÍAS DE COMERCIO ELECTRÓNICO**
  - UNSPSC (http://www.unspsc.org/)
  - RosettaNet (http://www.rosettanet.org/)
- **ONTOLOGÍAS MÉDICAS**
  - UMLS (http://www.nih.gov/research/umls/)
  - GALEN (http://www.opengalen.org/)
- **ONTOLOGÍAS DE BIOINFORMÁTICA**
  - GeneOntology (http://www.geneontology.org/)

5.8 Ejercicios resueltos

5.1. ¿Qué cambiaría y añadiría a la ontología de la Figura 5.8?

Respuesta:

- Supercomputador no es parte de ordenador, sino una subclase.
- No es recomendable omitir partes de los nombres, por ejemplo, en vez de "p tátil", el nombre del concepto debería ser "ordenador portátil".
- La ontología está reflejando una situación demasiado particular, en la que l portátiles son negros, y los PCs y los supercomputadores, negros.
- La ontología no está reflejando los aspectos esenciales de los conceptos que se tán describiendo. Por ejemplo, las características del procesador, de la memor etc, son importantes, y no aparecen.
- El primer axioma de la ontología es una tautología.
- Hay una contradicción entre los otros dos axiomas. Mientras que el primero d que un ordenador portátil tiene que pesar menos de 6 kg el segundo afirma q existe un portátil de 9 kg.

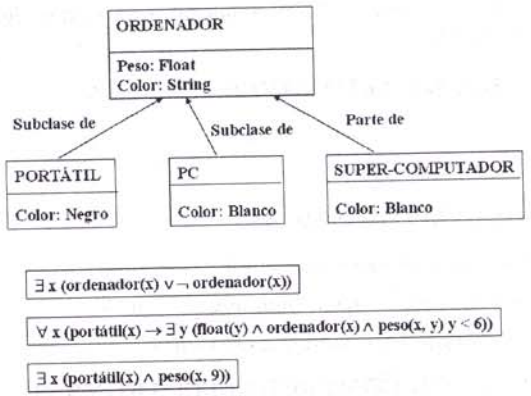

Figura 5.8: Ontología a evaluar.

- En caso de que se desee extender la ontología, hay varias alternativas (no excluyentes), por ejemplo:
  1. Creando ontologías sobre componentes de ordenadores.
  2. Integrando alguna ontología de unidades de medida para que el peso, por ejemplo, no tenga que estar expresado como un float, sino como un peso, concepto definido en otra ontología.
  3. Integrando alguna ontología de colores. Esta opción sólo es adecuada si el color se va a considerar una característica importante.
  4. Integrando alguna ontología con conceptos abstractos (objeto físico, característica, etc.) para afinar más el significado de los términos de la ontología de ordenadores.

■

5.9 Ejercicios propuestos

5.1. Utilizando WebODE, conceptuar, según METHONTOLOGY, una ontología de coches y después implementarla en OWL. Definir los términos que aprecen a continuación:

- Car.
- Sports car.
- Van.
- Berline.

- Horsepower.
- Maximum speed.
- Gasoline consumption.
- Type of brakes.
- Dimensions (length, high, etc.)

5.2. Importar, de nuevo con WebODE, la ontología OWL construida en el ejercicio 1. ¿Se ha perdido algún conocimiento? En caso negativo, ¿hay algún caso en que se pudiera haber perdido?

5.3. Cargar en Protégé la ontología OWL elaborada en el ejercicio 1. ¿Ha tenido algún problema?

5.4. Tomar cualquiera de las ontologías de Protégé e intente importarla desde WebODE.

5.5. ¿Cómo podría utilizar la ontología del ejercicio 1 para desarrollar un sistema Web que ayude a la compra-venta de vehículos?

Referencias

ARPÍREZ, J.C.; CORCHO, O.; FERNÁNDEZ-LÓPEZ, M. y GÓMEZ-PÉREZ, A.: «WebODE in a nutshell». *AI Magazine*, 2003, **24(3)**, pp. 37–47.

BARISH, G.; KNOBLOCK, C.A.; CHEN, Y.S.; MINTON, S.; PHILPOT, A. y SHAHABI, C.: «The theaterloc virtual application». En: R. Engelmore y H. Hirsh (Eds.), *Procceedings of the 12th Annual Conference on Innovative Applications of Artificial Intelligence (IAAI-00)*, pp. 980–987. Austin, Texas, 2000.

BECHHOFER, S.; HORROCKS, I.; GOBLE, C. y STEVENS, R.: «OilEd: a reasonable ontology editor for the Semantic Web». En: F. Baader; G. Brewka y T. Eiter (Eds.), *Procceedings of the oint German/Austrian conference on Artificial Intelligence (KI?01) (Lecture Notes in Artificial Intelligence LNAI 2174)*, pp. 396–408. Springer-Verlag, Berlin, Germany, Vienna, Austria., 2001.

BENTON, J. y KAMBHAMPATI, S.: «Toward Distributed Use of Large-Scale Ontologies». En: A. Farquhar; M. Gruninger; A. Gómez-Pérez; M. Uschold y P. van der Vet (Eds.), *Procceedings of the AAAI'97 Spring Symposium on Ontological Engineering*, pp. 138–148. Stanford University, California, 1997.

BENTON, J. y KAMBHAMPATI, S.: «OntoEdit: Collaborative Ontology Engineering for the Semantic Web». En: I. Horrocks y J.A. Hendler (Eds.), *Proceedings of the First International Semantic Web Conference (ISWC'02) (Lecture Notes in Computer Science LNCS 2342)*, pp. 221–235. Springer-Verlag, Berlin, Germany, Sardinia, Italy, 2002.

BERNARAS, A.; LARESGOITI, I. y CORERA, J.: «Building and reusing ontologies for electrical network applications». En: W. Wahlster (Ed.), *Procceedings of the European Conference on Artificial Intelligence (ECAI'96)*, pp. 298–302. John Wiley and Sons, Chichester, United Kingdom, Budapest, Hungary, 1996.

BERNERS-LEE, T.: *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by its Inventor*. HarperCollins Publishers, New York, 1999.

BRAY, T.; PAOLI, J.; SPERBERG-MCQUEEN, C.M. y MALER, E.: «Extensible Markup Language (XML) 1.0. W3C Recommendation». *Informe técnico*, W3C, 2000. Disponible en `http://www.w3.org/TR/PR-rdf-schema`.

BRICKLEY, D. y GUHA, R.V.: «RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation». *Informe técnico*, W3C, 2004. Disponible en `http://www.w3.org/TR/PR-rdf-schema`.

BRUIJN, J. DE: «The Web Service Modeling Language WSML. Deliverable D16.1v0.21.» *Informe técnico*, ESSI WSMO working group, 2006. Disponible en `http://www.wsmo.org/TR/d16/d16.1/v0.21/`.

CHAUDHRI, VK; FARQUHAR, A; FIKES, R; KARP, PD y RICE, JP: «Open Knowledge Base Connectivity 2.0.3». *Informe técnico*, Artificial Intelligence Center. SRI International, 1998. Disponible en `http://www.ai.sri.com/ okbc/okbc-2-0-3.pdf`.

CHEN, P.P.: «The Entity-Relationship Model: Toward a Unified View of Data». *ACM Transactions on Database Systems*, 1976, **1(1)**, pp. 9–36.

CORCHO, O.; FERNÁNDEZ-LÓPEZ, M. y GÓMEZ-PÉREZ, A.: «Ontological Engineering: Principles, Methods, Tools and Languages». En: C. Calero; F. Ruiz y M. Piattini (Eds.), *Ontologies for Software Engineering and Technology*, Springer-Verlag 2007.

DEAN, M. y SCHREIBER, G.: «OWL Web Ontology Language Reference. W3C Recommendation». *Informe técnico*, W3C, 2004. Disponible en `http://www.w3.org/TR/owl-ref/`.

DECLERCK, T. y USZKOREIT, H.: «State of the art on multilinguality for ontologies, annotation services and user interfaces. Esperonto deliverable D1.5». *Informe técnico*, Esperonto Project, 2003. Disponible en `http://www.esperonto.net`.

FARQUHAR, A.; FIKES, R. y RICE, J.: «The Ontolingua Server: A Tool for Collaborative Ontology Construction». *International Journal of Human Computer Studies* 1997, **6**, pp. 707–727.

FENSEL, D.: *The knowledge acquisition and representation language KARL*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.

FERNÁNDEZ-LÓPEZ, M. y GÓMEZ-PÉREZ, A.: «Overview and analysis of methodologies for building ontologies». *The Knowledge Engineering Review*, 2002, **17(2)** pp. 129–156.

FERNÁNDEZ-LÓPEZ, M.; GÓMEZ-PÉREZ, A. y JURISTO, N.: «METHONTOLOGY From Ontological Art Towards Ontological Engineering». En: *Procceedings of the Spring Symposium on Ontological Engineering of AAAI*, pp. 33–40. Stanford University, California, 1997.

FERNÁNDEZ-LÓPEZ, M.; GÓMEZ-PÉREZ, A.; PAZOS, A. y PAZOS, J.: «Building a Chemical Ontology Using Methontology and the Ontology Design Environment» *IEEE Intelligent Systems & their applications*, 1999, **4(1)**, pp. 37–46.

FIKES, R.; HAYES, P. y HORROCKS, I.: «OWL-QL: A Language for Deductive Query Answering on the Semantic Web». *Informe técnico*, KSL, 2003.

GANGEMI, A.; PISANELLI, D.M. y STEVE, G.: «An Overview of the ONIONS Project: Applying Ontologies to the Integration of Medical Terminologies». *Data & Knowledge Engineering*, 1999, **31(2)**, pp. 183–220.

GENESERETH, M.R. y FIKES, R.E.: «Knowledge Interchange Format. Version 3.0. Reference Manual». *Informe técnico Logic-92-1*, Computer Science Department. Stanford University, California, 1992. Disponible en `http://meta2.stanford.edu/kif/Hypertext/kif-manual.html`.

GRUBER, T.R.: «Ontolingua: A Mechanism to Support Portable Ontologies». *Informe técnico KSL-91-66*, Knowledge Systems Laboratory, Stanford University, Stanford, California, 1992. Disponible en `ftp://ftp.ksl.stanford.edu/pub/KSL_Reports/KSL-91-66.ps`.

GRUBER, T.R.: «A translation approach to portable ontology specification». *Knowledge Acquisition*, 2006, **5(2)**, pp. 199–220.

GRÜNINGER, M. y FOX, M.S.: «Methodology for the design and evaluation of ontologies». En: D. Skuce (Ed.), *Proccedings of the IJCAI95 Workshop on Basic Ontological Issues in Knowledge Sharing*, pp. 6.1–6.10, 1995.

GUARINO, N.: «Formal Ontology in Information Systems». En: N. Guarino (Ed.), *Procceedings of the 1st International Conference on Formal Ontology in Information Systems (FOIS'98)*, pp. 3–15. IOS Press, Amsterdam, 1998.

GUARINO, N.; MASOLO, C. y VETERE, G.: «OntoSeek: Content-Based Access to the Web». *IEEE Intelligent Systems & their applications*, 2006, **14(3)**, pp. 70–80.

GUARINO, N. y WELTY, C.: «Evaluating Ontological Decisions with OntoClean». *Communications of the ACM 45(2)*, 2002, **45(2)**, pp. 61–65.

GÓMEZ-PÉREZ, A.: «Some Ideas and Examples to Evaluate Ontologies». *Informe técnico*, Knowledge Systems Laboratory, Stanford University, California, 1994. Disponible en http://www-ksl.stanford.edu/KSL_Abstracts/KSL-94-65.html.

GÓMEZ-PÉREZ, A.: «Evaluation of Ontologies». *International Journal of Intelligent Systems 16(3)*, 2006, **16(3)**, pp. 391–409.

GÓMEZ-PÉREZ, A.; FERNÁNDEZ-LÓPEZ, M. y CORCHO, O.: *Ontological Engineering.* Springer, London, United Kingdom, 2003.

GÓMEZ-PÉREZ, A.; JURISTO, N.; MONTES, C. y PAZOS, J.: *Ingeniería del Conocimiento: Diseño y Construcción de Sistemas Expertos.* Ceura, Madrid, Spain, 1997.

GÓMEZ-PÉREZ, A. y ROJAS, M.D.: «Ontological Reengineering and Reuse». En: D. Fensel y R. Studer (Eds.), *Procceedings of the 11th European Workshop on Knowledge Acquisition, Modeling and Management (EKAW?99) (Lecture Notes in Artificial Intelligence LNAI 1621)*, pp. 139–156. Springer-Verlag, Berlin, Germany, Dagstuhl Castle, Germany., 1999.

HERMENEGILDO, M.; BUENO, F.; CABEZA, D.; CARRO, M.; GARCÍA, M.; LÓPEZ, P. y PUEBLA, G.: «The Ciao Logic Programming Environment». En: J.W. Lloyd; V. Dahl; U. Furbach; M. Kerber; K. Lau; C. Palamidessi; L.M. Pereira; Y. Sagiv y

P.J. Stuckey (Eds.), *Proccedings of the International Conference on Computationa Logic (CL'00) (Lecture Notes in Computer Science LNCS 1861)*, Springer-Verlag Berlin, Germany, London, United Kingdom, 2000.

HORROCKS, I.; FENSEL, D.; VAN HARMELEN, F.; DECKER, S.; ERDMANN, M. ; KLEIN, M.: «OIL in a Nutshell». En: R. Dieng y O. Corby (Eds.), *Proceeding of the 12th International Conference in Knowledge Engineering and Knowledg Management (EKAW'00) (Lecture Notes in Artificial Intelligence LNAI 1937)*, pp 1–16. Springer-Verlag, Berlin, Germany, Juan-Les-Pins, France, 2000.

IEEE: *Domain-Independent Temporal Planning in a Planning-Graph-Based Ap proach.* IEEE Computer Society. New York, 1996.

KALYANPUR, A.; PARSIA, B.; SIRIN, E.; GRAU, B.C. y HENDLER, J.: «Swoop: A Web Ontology Editing Browser». *Web Semantics: Science, Services and Agents o the World Wide Web*, 2006, **4(2)**, pp. 144–153.

KARP, P.D.; CHAUDHRI, V. y THOMERE, J.: «XOL: An XML-Base Ontology Exchange Language. Version 0.3». *Informe técnico*, Ar tificial Intelligence Center. SRI International, 1999. Disponible e `http://www.ai.sri.com/ pkarp/xol/xol.html`.

KIETZ, J.U.; MAEDCHE, A. y VOLZ, R.: «A Method for Semi-Automatic Onto logy Acquisition from a Corporate Intranet». En: N. Aussenac-Gilles; B. Biébo y S. Szulman (Eds.), *Procceedings of the EKAW'00 Workshop on Ontologies an Texts. CEUR Workshop Proceedings 51*, pp. 4.1–4.14. Juan-Les-Pins, Franc, 200 Disponible en `http://CEUR-WS.org/Vol-51/`.

KIFER, M.; LAUSEN, G. y WU, J.: «Logical Foundations of Object-Oriented an Frame-Based Languages». *Journal of the ACM*, 1995, **42(4)**, pp. 741–843.

KNUBLAUCH, H.; FERGERSON, R.; NOY, N.F. y MUSEN, M.A.: «The Protege OWL Plugin: An Open Development Environment for Semantic Web Applications». En S.A. McIlraith y D. Plexousakis (Eds.), *Procceedings of the 3rd International Se mantic Web Conference (ISWC'04) (Lecture Notes in Computer Science LNC 3298)*, pp. 229–243. Springer-Verlag, Berlin, Germany, Hiroshima, Japan, 2004.

KOIVUNEN, M.R.: «Web Tagging with Annotea Shared/Social Bookmarks an Topics». En: *Procceedings of the Tagging Workshop at the World Wide We Conference*, pp. 23–25. Edinburgh, United Kingdom, 2006. Disponible e `http://annotea.org/www2006/annotea.htm`.

LASSILA, O. y SWICK, R.: «Resource Description Framework (RDF) Model an Syntax Specification (W3C Recommendation)». *Informe técnico*, W3C, 2000. Disponible en `http://www.w3.org/TR/REC-rdf-syntax/`.

LENAT, D.B. y GUHA, R.V: *Building Large Knowledge-Based Systems.* Addison-Wesley, 1990.

LOWE, E.J.: *The Four-Category Ontology: A Metaphysical Foundation for Natural Science.* Oxford University Press, 2006.

LUKE, S y HEFLIN, JD: «SHOE 1.01. Proposed Specification». *Informe técnico*, Parallel Understanding Systems Group. Department of Computer Science. University of Maryland, 2000. Disponible en `http://www.cs.umd.edu/projects/plus/SHOE/spec1.01.htm`.

LÉGER, A.; ARBANT, G.; BARRETT, P.; GITTON, S.; GÓMEZ-PÉREZ, A.; HOLM, R.; LEHTOLA, A.; MOUGENOT, I.; NISTAL, A.; VARVARIGOU, T. y VINESSE, J.: «MKBEEM: Ontology domain modeling support for multilingual services in e-Commerce». En: *Proceedings of the ECAI'00 Workshop on Applications of Ontologies and PSMs*, pp. 19.1–19.4. Berlin, Germany, 2000.

MACGREGOR, R.: «Inside the LOOM clasifier». *SIGART bulletin*, 1991, **2(3)**, pp. 70–76.

MAEDCHE, A.; MOTIK, B.; STOJANOVIC, L.; STUDER, R. y VOLZ, R.: «Ontologies for Enterprise Knowledge Management». *IEEE Intelligent Systems*, 2003, **18(2)**, pp. 26–33.

MENA, E.; KASHYAP, V.; SHETH, A.P. y ILLARRAMENDI, A.: «OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies». En: W. Litwin (Ed.), *Proceedings of the First IFCIS International Conference on Cooperative Information Systems (CoopIS'96)*, pp. 14–25. Brussels, Belgium, 1996.

MOTTA, E.: *Reusable Components for Knowledge Modelling: Principles and Case Studies in Parametric Design.* IOS Press, Amsterdam, The Netherlands, 1999.

NECHES, R.; FIKES, R.E.; FININ, T.; GRUBER, T.R.; SENATOR, T. y SWARTOUT, W.R.: «Enabling technology for knowledge sharing». *AI Magazine*, 1991, **12(3)**, pp. 36–56.

NOY, N.F.: «Evaluation by Ontology Consumers». *IEEE Intelligence Systems*, 2006, **19(4)**, pp. 80–81.

NOY, N.F.; FERGERSON, R.W. y MUSEN, M.A.: «The knowledge model of Protege-2000: Combining interoperability and flexibility». En: R. Dieng y O. Corby (Eds.), *Procceedings of the 12th International Conference in Knowledge Engineering and Knowledge Management (EKAW'00) (Lecture Notes in Artificial Intelligence LNAI 1937)*, pp. 17–32. Springer-Verlag, Berlin, Germany, Juan-Les-Pins, France, 2000.

NOY, N.F. y MUSEN, M.A.: «PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment». En: P. Rosenbloom; H.A. Kautz; B. Porter; R. Dechter; R. Sutton y V. Mittal (Eds.), *Procceedings of the 17th National Conference on Artificial Intelligence (AAAI'00)*, pp. 450–455. Austin, Texas, 2000.

OMG: «Ontology Definition Metamodel. Third Revised Submission to OMG/ RFP ad/2003-03-40». *Informe técnico*, Object Management Group, 2005. Disponible en `http://www.omg.org/docs/ad/05-08-01.pdf`.

RAGGETT, D.; HORS, A. LE y JACOBS, I.: «HTML 4.01 Specification. W3C Recommendation». *Informe técnico*, W3C, 1999. Disponible en `http://www.w3.org/TR/html401/`.

RUMBAUGH, J.; JACOBSON, I. y BOOCH, G.: *The Unified Modeling Language Reference Manual.* Addison-Wesley, Boston, Massachusetts, 1998.

STAAB, S.; SCHNURR, H.P.; STUDER, R. y SURE, Y.: «Knowledge Processes and Ontologies». *IEEE Intelligent Systems*, 2001, **16(1)**, pp. 26–34.

STEVE, G.; GANGEMI, A. y PISANELLI, D.M.: «Integrating Medical Terminologies with ONIONS Methodology». En: *Kangassalo H, Charrel JP (eds) Information Modeling and Knowledge Bases VIII*, IOS Press, Amsterdam, The Netherlands, 1998. Http://ontology.ip.rm.cnr.it/Papers/onions97.pdf.

STUCKENSCHMIDT, H.: «Using OIL for Intelligent Information Integration». En: V.R. Benjamins; A. Gómez-Pérez y N. Guarino (Eds.), *Procceedings fo the ECAI'00 Workshop on Applications of Ontologies and Problem Solving Methods*, pp. 9.1–9.10. Berlin, Germany, 2000.

STUDER, R.; BENJAMINS, V.R. y FENSEL, D.: «Knowledge Engineering: Principles and Methods». *IEEE Transactions on Data and Knowledge Engineering*, 1998, **25(1-2)**, pp. 161–197.

STUMME, G. y MAEDCHE, A.: «FCA-MERGE: Bottom-Up Merging of Ontologies». En: N. Bernhard (Ed.), *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI 2001)*, pp. 225–234. Morgan Kaufmann Publishers, San Francisco, California, Seattle, Washington, 2001.

USCHOLD, M. y GRÜNINGER, M.: «Ontologies: Principles, Methods and Applications». *Knowledge Engineering Review*, 1996, **11(2)**, pp. 93–155.

VAN HARMELEN, F.; PATEL-SCHNEIDER, P. F. y HORROCKS, I.: «Reference Description of the DAML+OIL (March 2001) Ontology Markup Language». *Informe técnico*, The DARPA Agent Markup Language Homepage, 2001. Disponible en `http://www.daml.org/2001/03/reference.html`.

WELTY, C. y GUARINO, N.: «Supporting Ontological Analysis of Taxonomic Relationships». *Data and Knowledge Engineering*, 2001, **39(1)**, pp. 51–74.